:codica

Тестове завдання Node.js Software Engineer

Мета

Розробити API сервер, використовуючи можливості Node.js з використанням фреймворку **NestJS.**

Ідея

Додаток, що розробляється - фінансовий менеджер, що допомагає вести облік фінансів власника.

Задача

Розробити API для побудови веб та мобільного додатків.

- У додатку є Банки (Bank), Транзакції (Transaction) та Категорії Транзакцій (Category).
- У додатку можна додати нові банки та категорії транзакцій.
- Транзакції надходять на окремий Webhook.
- У транзакції може бути більше однієї категорії.
- Видалення банків та категорій можливе лише за відсутності транзакцій у них.
- Можливе вилучення транзакцій.
- Баланс у банку повинен перераховуватись при додаванні або видаленні транзакцій (або бути розрахованим на основі наявних транзакцій).

Схема вихідних даних

Bank

- id
- name
- balance
- …

Transaction

- id
- amount

- type (profitable | consumable)
- …

Category
- id
- name
- …

Потрібно реалізувати API методи, які дозволять:

- Bank: створити, видалити, отримати один, отримати все, відредагувати
- Transaction: створити (webhook), видалити, отримати все
    - Для отримання всіх транзакцій необхідно реалізувати пагінацію
    - Для webhook, за бажанням, можна додати api key authorization
- Category: створити, видалити, отримати одну, отримати все, відредагувати
- Метод отримання статистики
    - Приймає categoryIds, fromPeriod, toPeriod
    - Повертає об'єкт у форматі { ім'я категорії: баланс }, наприклад: { їжа: -2500, зарплата: +12000, бензин: -700}

Технічні вимоги

- Node.js
- NestJS
- TypeScript
- PostgreSQL
- Docker
- Type ORM
- Swagger API docs

На виході має вийти проєкт на Github.com **з інструкцією із запуску та документацією до ендпоїнтів**. Додаток **повинен** запускатися в **Docker.**

За можливості створити **Github Page для Swagger документації.**

Бонусні завдання

Ці завдання не оцінюються в першу чергу, але матимуть вагу за наявності рівнозначних схожих реалізацій:
- Покрити функціональність API сервера тестами (Unit, Request/Integration)
- Реалізувати Логування
- Реалізувати Глобальну обробку помилок

:codica

Після реалізації

- При направленні ТЗ на перевірку, необхідно додатково надати інформацію:
  - Чи використовували ви якісь додаткові бібліотеки? Якщо так, то навіщо?
  - Скільки приблизно часу зайняло виконання тестового?
  - Якби треба було вивести цей проєкт у продакшн, що б ви покращили?